NASA/TM-1999-206588

Automated Testing Experience of the Linear Aerospike SR-71 Experiment (LASRE) Controller

Richard R. Larson
NASA Dryden Flight Research Center
Edwards, California

September 1999

The NASA STI Program Office . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the lead center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA's counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or cosponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and mission, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that complement the STI Program Office's diverse offerings include creating custom thesauri, building customized databases, organizing and publishing research results . . . even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at ***http://www.sti.nasa.gov***

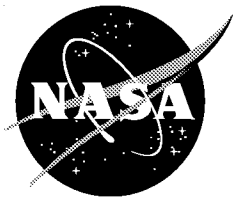
- E-mail your question via the Internet to help@sti.nasa.gov

- Fax your question to the NASA Access Help Desk at (301) 621-0134

- Telephone the NASA Access Help Desk at (301) 621-0390

- Write to:
  NASA Access Help Desk
  NASA Center for AeroSpace Information
  7121 Standard Drive
  Hanover, MD 21076-1320

NASA/TM-1999-206588



Automated Testing Experience of the Linear Aerospike SR-71 Experiment (LASRE) Controller

Richard R. Larson
NASA Dryden Flight Research Center
Edwards, California

National Aeronautics and
Space Administration

Dryden Flight Research Center
Edwards, California 93523-0273

September 1999

NOTICE

Use of trade names or names of manufacturers in this document does not constitute an official endorsement of such products or manufacturers, either expressed or implied, by the National Aeronautics and Space Administration.

Available from the following:

NASA Center for AeroSpace Information (CASI)
7121 Standard Drive
Hanover, MD 21076-1320
(301) 621-0390

National Technical Information Service (NTIS)
5285 Port Royal Road
Springfield, VA 22161-2171
(703) 487-4650

Automated Testing Experience of the Linear Aerospike SR-71 Experiment (LASRE) Controller

Richard R. Larson
NASA Dryden Flight Research Center
Edwards, California

Abstract

System controllers must be fail-safe, low cost, flexible to software changes, able to output health and status words, and permit rapid retest qualification. The system controller designed and tested for the aerospike engine program was an attempt to meet these requirements. This paper describes (1) the aerospike controller design, (2) the automated simulation testing techniques, and (3) the real time monitoring data visualization structure. Controller cost was minimized by design of a single-string system that used an off-the-shelf 486 central processing unit (CPU). A linked-list architecture, with states (nodes) defined in a user-friendly state table, accomplished software changes to the controller. Proven to be fail-safe, this system reported the abort cause and automatically reverted to a safe condition for any first failure. A real time simulation and test system automated the software checkout and retest requirements. A program requirement to decode all abort causes in real time during all ground and flight tests assured the safety of flight decisions and the proper execution of mission rules. The design also included health and status words, and provided a real time analysis interpretation for all health and status data.

Nomenclature

ACTS	aerospike controller test system
ARM/H_2	dual button on control panel to first, arm autosafe modes and second, dump hydrogen
AS	autosafe state
ASC	allied signal controller
ASSC	aerospike system controller
CCP	cockpit control panel
CIMS	calibration information management system
CO	cutoff state
cp	cockpit panel simulator page
CPDF	control program data file
CPU	central processing unit
FDAS	flight data access system
GH_2	gaseous hydrogen
GHe	gaseous helium

He	helium gas
H_2	hydrogen gas
H_2O	water, and label for control panel button to dump water
ID	identification
I/O	input/output
imxtst	time out parameter, ms
LASRE	Linear Aerospike SR-71 Experiment
LO_2	liquid oxygen, and label for control panel button to dump liquid oxygen
logc	log clear command
MAH	master abort hold state
MAS	master abort sequence state
MCC	mission control center
mdl	model simulation page
OFP	operational flight program
PCM	pulse code modulator
prst	power reset
psia	pounds per square inch, absolute
RLV	reusable launch vehicle
RTF	real time FORTRAN
setredline	abort limit value
SMART	signal management for analysis in real time
SS	start state
TEA-TEB	triethylaluminum-triethylboron
TRAPS	telemetry and radar acquisition processing system
TM	telemetry
V and V	verification and validation
val	value simulation page
VME	Versa Module Eurocard
waitst	wait state

Introduction

Efforts in the space industry are being made to reduce the cost of placing payloads into orbit. One proposed design is a single-stage-to-orbit approach, which utilizes a rocket-powered reusable launch vehicle (RLV). For this concept to be feasible, a more efficient propulsion system must be explored. One such system is the linear aerospike engine, which was first developed in the 1960's.[1–3] As a flight demonstrator Lockheed Martin is developing the X-33 program to validate the feasibility of the RLV

concepts and the aerospike engine. To support this program the Linear Aerospike SR-71 Experiment (LASRE) [4] was initiated to obtain flight test data on this unique engine design.

An SR-71 aircraft was modified to carry the aerospike engine and its feed systems. These modifications were mounted at the rear of the aircraft between the ventrals as shown in figure 1. The Linear Aerospike SR-71 Experiment (LASRE) engine supply systems were enclosed in a pod consisting of hydrogen (H_2) for the fuel, liquid oxygen (LO_2) for the oxidizer, water (H_2O) for engine cooling, triethylaluminum-triethylboron (TEA-TEB) for the igniter and helium (He) for purging and tank pressurization (fig 2). Control and monitoring of these systems was accomplished using a single channel, aerospike system controller (ASSC).

The basic function of the ASSC is operation of solenoid valves that control the LASRE engine supply systems. A subset of the valve layout is shown in figure 3. The ASSC manipulation of these valves produces a firing-flow sequence as shown in figure 4. The start-states (SS) sequence begins with purges of helium (He) in the LO_2 and TEA-TEB lines. A liquid oxygen trickle flow is then commanded to prechill the line, which is followed by an initial water-trickle flow to fill the passages in the engine. The main water and liquid oxygen flow starts and then the TEA-TEB igniter begins. The hydrogen line is purged briefly to prevent backflow and finally, the valve is opened. At this point the engine firing begins and lasts for 3 sec. The cutoff (CO) states begin when the hydrogen and liquid oxygen valves are closed. The water flow stops, and then purges begin to prevent backflow. The autosafe (AS) modes would be initiated next to complete water, liquid oxygen, and hydrogen purges.

The LASRE program was conceived with an aggressive schedule to obtain flight data at minimum cost. Design considerations for the ASSC were low-cost, man-rated, reliable, fail-safe, with easy to change software, and rapid verification and validation (V and V) capabilities of software. Any first failure needed to be immediately detected by the system and followed by an automatic abort to a safe configuration. Testing configurations and anomalies frequently required changes to the ASSC software. Software modifications had to be quickly and thoroughly tested to meet schedule constraints. Any aborts needed to be identified in real time for safety of flight decisions and be quickly debugged. These challenging requirements led to the three main parts of this paper. First, a description of the ASSC fail-safe, single-string, design; second, the automated V and V software testing; and third, the real time, intelligent monitoring system used for LASRE system tests. Use of trade names or names of manufacturers in this document does not constitute an official endorsement of such products or manufacturers, either expressed or implied, by the National Aeronautics and Space Administration.

LASRE Control System

The LASRE controller architecture is comprised of four primary components (fig 5). These are the ASSC, the allied signal controller (ASC), the pulse code modulator (PCM) data instrumentation system, and the cockpit control panel (CCP). The solenoid valves are controlled from the ASSC, which receives system temperature and pressure data from the PCM instrumentation system and operator commands from the CCP. Command of the main control valves for the water, liquid oxygen, and hydrogen feed systems is done from the ASC, which receives inputs from the ASSC. Health and status information is also passed from the controllers and telemetered (TM) to the ground.

Figure 6 shows the LASRE cockpit control panel, which provides the operator with status lights and push buttons for mode control. The firing sequence starts by first pressing the PRESTART and then the START button. After the firing, the autosafe mode is armed by pressing the ARM/H_2 button. From this mode any of the autosafe modes may be selected to dump the hydrogen, water, and liquid oxygen systems.

Aerospike system controller (ASSC)

Software modules for the ASSC are shown in figure 7. The ASSC software consists of the operational flight program (OFP) and the control program data file (CPDF). A state table is merged with the calibration information management system (CIMS) data file to create an executable state table load file. The OFP normally does not need to be changed. These software modules will now be described in more detail.

Operational flight program (OFP)

The OFP manages all controller input/output (I/O) functions. It reads the experiment pressures and temperatures from the PCM data stream, the ASC status words, and switches from the CCP. Outputs include solenoid valve commands; main control valve motor clutch (or motor relay) commands; inputs to the ASC for operation of the hydrogen, liquid oxygen, and water valves; and CCP lights.

The OFP begins execution at power up. At initialization the executable state table file is opened. Dynamic memory allocation is created for the states, transducers, and PCM variables. The decommutator card is initialized with data from the setup record and fixed data. Each record is read in the file and the data is passed through to the state in which it belongs. The watchdog, PCM, and ASC interrupts are initialized and monitored.

Failure monitoring is performed from continuous self-tests:

- PCM mismatch wrap test
- watchdog monitor
- over-temperature abort (controller, main servoamps)
- ASC word check (count, status)
- main valve command and position mismatch

A PCM interrupt is issued every 2.5 ms as shown in the real time loop description in figure 8. In frame 0 the controller reads the input data and in frame 2 the controller compares a copy of the data from frame 0. If the comparison test fails, then a fail is declared for the PCM mismatch wrap test and the abort flag is set. There is a 1-frame persistence counter set for this and all other failures.

State table

The state table defines actions and redline/transition checks that the controller must follow to perform a test point instruction. It is written in a readable format to aid the user in modifying the instructions. The state table contains the following: version (identifier for the table), state declarations (functions for each state), and an end (end of table). Each state structure must contain a state name, a default state name, an actions list, and a transition function. A state table example for typical components is shown in figure 9. When this state is entered, the controller performs the actions listed and stays at this point until the timer reaches 2.0 sec, at which point the transition to SS1:30 is done. The state table commands for the actions and transition functions are shown in tables 1 and 2. These functions are used to control the LASRE systems as required for the test.

Control program data file (CPDF)

The CPDF reads the CIMS data file and the state table. The CIMS file contains the calibrations for all the PCM signals. An executable program is created from these two files. The CPDF is written in C programming language and interprets the state table as a linked-list data structure. This is illustrated in figure 10. Generally at every state there are two pointer possibilities, (1) normal transition and (2) abort transition. Memory allocation is created for each state defined in the state table and for every transducer that is referenced including its calibration coefficients. Any number of state transition points may be easily added or deleted from the state table, thereby making this structure easy to build.

Transducer records are created from the CIMS file by searching for frame word position, frame number, frame depth, data type, minimum count, maximum count, and scaling coefficients. The state table is read line-by-line with the appropriate records such as state, default, action, transition, and end. The CIMS data parameter values are converted from engineering units back to counts for the real time processing.

Flow chart

The states and the transitions flow is represented in figure 11 as a flow chart. With power-on the system starts at the initialization states and automatically transitions to a master standby state. The system waits at this mode until commanded to prestart or to the autosafe modes for a normal start path. From prestart the system may be commanded to start, which is then followed by an automatic transition to shutdown and then return back to a master standby mode. At each state there is an automatic abort path possible. The autosafe modes may be entered from master standby or master abort hold states as shown.

Redlines

Parameter redlines are used to set limits, either above or below specific signals, to test for abnormal conditions. This may be indicative of a stuck valve, failed sensor, leak, or plugged line. Signal redlines are set or cleared in any states, as desired. This allows for a flexible system. Table 3 shows the fault transitions, including redline values, for each state.

Status words

The controller status words are formatted on the PCM data stream for recording and monitoring. The state transition information is shown in table 4. These words provide information about the current state number, type of transition, redline values in effect, and abort causes. Additional abort information is contained in a *word* in which bits are defined in table 5. These *words* contain all controller abort information.

Allied signal controller (ASC)

The ASC receives commands from the ASSC for the main valves (GH_2, LO_2, and H_2O). It then provides detailed control for these motor-operated valves, using position feedback for the LO_2 and H_2O valves to provide on/off ramping and pressure feedback for the GH_2 valve for pressure regulation. The ASC also receives status requests from and provides health/status information to the ASSC. It is capable of shutting the main valves and stopping the test if it detects an error in the status requests from the ASSC. This feature provides a measure of control system abort redundancy.

Pulse code modulator (PCM)

The PCM data instrumentation system collects and packages all LASRE data into frames, in a continuously cycling data stream, for telemetry to the ground. This includes experimental data, control system temperature and pressure data, and ASSC status information. The entire data stream is also provided to the ASSC, which uses control system pressures and temperatures. The timing of these data frames also provides external interrupts for the ASSC.

Aerospike Controller Test System Description (ACTS)

The ASSC software was modified and tested using a real time simulation. A simplified block diagram of the LASRE controller simulation is shown in figure 12. This test system allowed for closed-loop testing with the controller by driving all the necessary hardware interfaces. A 486 CPU was used to create an executable state table from a CIMS signal calibration file and a source state table file. This executable file was then downloaded into the ASSC for testing. As part of this simulation an Aerospike Controller Test System (ACTS)[5] was designed to facilitate V and V testing of the ASSC.

The ACTS allowed inputs to be controlled as a function of the state sequence so that the appropriate time intervals were maintained to allow the ASSC to proceed through the entire test sequence. In addition, simulated failure inputs were provided to verify that the controller would follow the proper abort sequences.

Sensor model

The experiment sensors consist of pressure values, temperature values, and an emergency shutdown switch voltage. These sensor values are output to the PCM data system through digital-to-analog converter boards in the Versa Module Eurocard (VME) card cage. Values are set for these sensors by the ACTS software that is based on the current state of the controller. This sensor model file is shown in table 6. As the state table begins execution sensor redlines are established according to the current state. The ACTS reads the controller states and sets the appropriate sensor values as defined in the table.

Pulse control modulator data system

The PCM data system reads the analog sensor values and generates a PCM bit stream for input to the ASSC. The PCM data system also inputs a serial data stream from the ASSC containing health and status information. The PCM bit stream is also sent to a PCM decommutator board in the VME card cage so that the ACTS program knows the controller state.

Pulse control modulator decommutator board

The PCM bit stream generated by the PCM data system includes the state of the controller. Since the ACTS program needs to know the controller state in order to set sensor values, this PCM bit stream must be read.

Solenoid valves

There are 19 solenoid valves that are controlled directly by the ASSC. Valve commands are input from the controller by way of an input discrete board in the VME card cage.

Allied signal controller (ASC)

The ACTS program normally simulates the ASC. However, there is provision to substitute the flight hardware ASC in the simulation if desired. The interface for the ASC includes 4 discrete inputs to the test system through an input discrete board in the VME card cage. Three of the discrete inputs are used by the controller to operate three valves (LO_2, GH_2, and H_2O). The remaining discrete is used to request status information. The ASSC status request discrete line is set high for approximately 2.5 ms to request status. The status request occurs at a 100 Hz rate. The ACTS program responds to the status request within approximately 5–7.5 ms, otherwise a fault bit is set.

Cockpit control panel (CCP)

The CCP signals are interfaced to the controller through output and input discrete boards in the VME card cage. A simple graphic display was generated to simulate the cockpit panel with the proper buttons, toggle switch, and colored lights. This graphic display was completely functional for the software tester.

Aerospike controller test system software (ACTS)

The ACTS program is written in FORTRAN and ANSI C. The user interface is the same as that used in the standard Dryden flight simulations. This includes a simple command line and display interface.

The real time part consists of two main loops. The primary loop runs at 100 Hz and includes the cockpit panel input/output, valve/clutch monitoring, and data recording. The secondary loop runs at a much higher rate (~1000 Hz). This fast loop checks the status request line and the PCM bit stream and responds with the appropriate action. When the status request line goes high, status information is immediately sent to the aerospike controller through the serial port. When a new PCM minor frame is received, the controller state is checked as well as the subframe identification (ID). If the controller state has been changed, the appropriate sensor values are updated based on the state table. The subframe ID determines which set of values to output for the multiplexed temperatures.

Manual ASSC testing

The ACTS program was designed to set and monitor the I/O to the ASSC. Verification and validation testing was conducted by starting the ACTS program and then powering up the controller. Both the ACTS program and the controller powered up in the first state indicated by the state table. From that point, testing was conducted manually, if desired, by pressing the appropriate buttons on the cockpit panel to initiate and proceed through the test sequence.

The ACTS program created a time-tagged, log file of the following items during the test:

1) Controller states

2) Transition status

3) Sensor value set (based on state change)

4) Valve/Clutch commanded open/close (on/off)

5) Cockpit buttons change position

6) Cockpit lights turn on/off

Automated ASSC Test Results

The ACTS program was designed to read a command file that would control the same simulation inputs as were done manually. This structure allowed for a repeatable, fast, and time-controlled test sequence. An example of a script file is shown in table 7. This script sets the pressure signal (PT0001) to 325 psi when the ASSC is executing state number SS1:36. At this moment the state table is commanding a pressure range test as shown below.

Setredline PT0001 outside 150 320 goto mas:1 Chamber Pr redline

A timeout parameter (imxtst) and wait state (lwaitst) is set in the script in case the test fails. For this test a 60-sec wait time is set. If the state does not go to the MAH:1 within 60 sec, the following message is set.

WARNING: Timed out waiting for state: MAH:001

If the signal is outside the range 150 to 320 psi, this test verifies that the setredline command for this pressure causes a transition to the specified abort state (mas:1). For this example only the upper limit of 320 only is tested. A portion of the resulting test log file that was generated for this script is shown in table 8. When state SS1:36 was entered, the signal PT0001 was set to 325 psi (exceeding the 320 psi limit). Other operations were performed as commanded for this state and finally a transition message (shown in bold type) was generated. This message was generated by reading the ASSC status words described in table 4. *Words* 1–6 identified the transition signal (PT0001), *word* 7 identified the code (O - outside), *word* 8 is not applicable for this cause, *word* 9 contains the upper limit checked (320 psi is 2078 PCM counts), *word* 10 has the current value (325 psi is 2110 PCM counts), and finally *words* 11–14 show the state which the transition occurred (SS1:36).

Hundreds of similar scripts are written to automate the V and V process for the ASSC. These scripts may be combined in a single, autotest file and all run at once. An excerpt of an autotest file is shown below.

```
# Run the script

macro /home/not/acts/ST_HOTFIRE/VnV/Scripts/Start/start_as01.scr

# Save the logfile

logs /home/not/acts/ST_HOTFIRE/VnV/Logfiles/Start/start_as01.log

# Run the script

macro /home/not/acts/ST_HOTFIRE/VnV/Scripts/Start/start_as02.scr

# Save the logfile

logs /home/not/acts/ST_HOTFIRE/VnV/Logfiles/Start/start_as02.log

# Run the script

macro /home/not/acts/ST_HOTFIRE/VnV/Scripts/Start/start_as03.scr

# Save the logfile

logs /home/not/acts/ST_HOTFIRE/VnV/Logfiles/Start/start_as03.log
```

Script files were created to test every normal/abort state transition, all controller functions, all redline aborts, and all aborts caused by the ASSC and ASC monitors. Refer to figure 11 for the state transitions and table 3 for the fault transitions and redline conditions. The status words themselves described in tables 4 and 5 were tested for correct status. As a result of this software testing automation, an exhaustive set of test cases were quickly executed and easily analyzed. The time to qualify software changes to support LASRE testing was significantly shortened.

Mission Control Center Real Time Monitoring Results

Intelligent monitoring systems have been developed for flight programs such as the space shuttle to aid in identifying problems in real time.[6] The LASRE program had a similar requirement to monitor and process the ASSC status words (tables 4 and 5) in real time so that any abort condition would be immediately known. Therefore, an automated, real time analysis tool was used to filter the status words based on events as textual strings and numerical values and output this information onto an X-window message stack. This tool was developed in-house at NASA Dryden and is called signal management for analysis in real time (SMART). The rule-based, intelligent real time monitor[7] was used for all ground and flight tests in the mission control center (MCC).

Real time FORTRAN (RTF) processing

The PCM real time data processing is shown in a top-level data flow in figure 13. Signals are telemetered at their frame rate. This is 100hz for the ASSC. The data enters a receiver rack, which sends it out to a telemetry and radar acquisition processing system (TRAPS), and front-end processor. The data is converted into engineering units in a real time FORTRAN (RTF) processor at 100 Hz and then sent out on an Ethernet data server. This server sends the data to the Unix workstations in the MCC at about 15hz. Unfortunately, this is too slow to read the abort status words that exist for only 10 ms. In order to catch the abort status words, latching logic was programmed into the RTF as shown in figure 14. The ASSC status words are read at 100hz and if there are no aborts several previous states are saved in a ring buffer. When an abort flag is set, the ring buffer stops updating and a latch flag is set. These buffered signals contain the abort cause and are sent out through the Ethernet server to the SMART application. The SMART decodes these raw buffered *words* into messages defining the abort reason. When the abort flag is reset, the ring buffer begins updating the status words again and the latch flag is reset.

SMART message stack

The SMART monitor application was used in the MCC to display the LASRE abort codes. An example of the message stack is shown in figure 15. New messages are added to the stack at the top and the older messages are pushed down. As messages are cleared off the stack, the messages automatically compress, thus filling in the gap between messages.

The example shown in the figure is from an ASSC abort. The cause was determined by reading the status words defined in table 4 from the latched ring buffer and converting this information to a message string. In this case the pressure signal (PT0651) exceeded a redline limit of 30 psi. The pressure at the time of the abort was 33 psi and the ASSC was in the state as 2:11. Other information about each of the main control valves is also shown.

SMART log file

The SMART message window also had an option to save all the messages that are both set and rescinded to a log file. Included in the SMART knowledge base were rules for the state messages.

After the test was completed, the SMART log file was saved and Unix commands were run to scan out the state messages. This was done to get a quick and rough estimate of which states were executed and their times. The flight data access system (FDAS) states and times had to be run the following day for exact states and times from a batch file. A comparison of the FDAS and SMART states is shown in table 9. The SMART missed some of the states because of the slow sample rate of the Ethernet server and the SMART architecture that writes to a file on the hard disk while still running in real time. The times were comparable, however, for a quick and gross check. This log of the state times proved invaluable for determining event times for batch data analysis and plotting.

Conclusion

The aerospike system controller (ASSC) had challenging requirements to be fail-safe, low cost, flexible to software changes, and a quick V and V software turnaround. These conditions were satisfied by using a clever architecture and through the use of auto-testing tools.

The ASSC was fail safe because of the conservative approach in actively monitoring status, health, and sensor redlines. Any failure in the system would automatically transition to an orderly and safe shutdown sequence

Using a simplex signal set from the pulse code modulator (PCM) rather than dedicated ASSC signals had never been done at NASA Dryden. This approach resulted in a significant cost reduction. Calibrations for PCM signals became an input file for the ASSC software. The operational flight program (OFP) interrupts were slaved to the PCM frames. The ASSC was single string, which also reduced the cost.

Software changes were easy to make because of the link-list structure. States could easily be added, deleted, or modified in the state table file. The control program data file (CPDF) software made it easy to change the calibration information management system (CIMS) and state table input files to generate a new executable ASSC program.

The aerospike controller test system simulation provided a test tool to automate the verification and validation runs. Exhaustive script files were run from a single master file to automatically test every path and abort. The log files provided an archive record of the test results.

Status words from the ASSC provided excellent insight to ascertain the condition of the controller. These words contained states, abort codes, transitions, and input/output discretes. This data proved invaluable in providing information about the controller to debug and assess problems. Status words were automatically decoded in real time using the signal management for analysis in real time (SMART) application in the MCC during all testing of the LASRE. Immediate analysis of any controller problem was demonstrated and proved to be a great help in conducting the LASRE tests. In addition, the automated time-tagged states that were generated from SMART in real time during the testing were a tremendous help in identifying event times for further off line analysis.

References

[1] Angelino, Giafranco, "Approximate Method for Plug Nozzle Design," *AIAA Journal*, vol 2, no. 10, Oct. 1975, pp. 1834-1835.

[2] Rockwell International Corporation, "Advanced Aerodynamic Spike Configurations: Volume 2–Hot Firing Investigations," AFRPL-TR-67-246, Sept. 1967. (Distribution authorized to U.S. Government agencies and their contractors; other requests shall be referred to WL/FIMS Wright-Patterson AFB, Ohio 45433–6503.)

[3] Mueller, T. J. and W. P. Sule, "Basic Flow Characteristics of a Linear Aerospike Nozzle Segment," ASME-72-WA/Aero-2, Nov. 1972.

[4] Corda, Stephen, Bradford A. Neal, Timothy R. Moes, Timothy H. Cox, Richard C. Monaghan, Leonard S. Voelker, Griffin P. Corpening, and Richard R. Larson, *Flight Testing the Linear Aeropike SR-71 Experiment (LASRE)*, NASA TM-1998-206567, Sept 1998.

[5] Kellogg, Gary V., and Ken A. Norlin, "Aerospike Controller Test System," *NASA Tech Briefs*, vol. 21 No.2, Feb 1997, pp.28–30.

[6] Land, Sherry A., Jane T. Matlin, Carrol Thronesberry, Debra L. Schreckenghost, *Making Intelligent Systems Team Players, A Guide to Developing Intelligent Monitoring Systems*, NASA TM 104807, July 1995.

[7] Larson, Richard, and D. Edward Millard, *A Rule-Based System for Real-Time Analysis of Control Systems*, NASA TM 104258, Oct 1992.

Table 1. State table actions functions.

Function	Description
ON <light discrete> OFF	Set cockpit control lights
OPEN <solenoid valve> CLOSE	Command to open/close solenoid valves
RESET TIMER	Reset state table timer
ACTIVATE <clutch> DEACTIVATE	Engage main valve clutches/relays
SETDEF <MAINSTATUSMON> <ALLIEDSTATUSMON>	Starts health monitoring of specified processor
SETREDLINE ABOVE, BELOW, OUTSIDE <pressure signal>goto <state>	The function transitions to the goto state when the pressure signal is outside the specified limits
SETREG<pressure signal> <lolim><hilim><solenoid valve>	The pressure is regulated from the upper limit to the lower limit by opening the specified valve
CLEARREG	Clears regulator function
SETPRES<pressure signal><lolim><hilim> <solenoid valve>	The pressure is regulated from the lower limit to the upper limit by opening the specified valve
CLEARPRESS	Clears pressure regulator function
CLEARABORT	Resets any abort flags

Table 2. State table transition functions.

Function	Description
TIMER wait <time> goto <state>	Transition to specified state after wait time (since last reset time function)
<pressure signal> ABOVE <value><goto><state>	Transition to specified state if specified pressure is above the value
<pressure signal> BELOW <value><goto><state>	Transition to specified state if specified pressure is below the value
ON goto <state>	Transition to specified state if the input discrete is true
OFF goto <state>	Transition to specified state if the input discrete is false
SAVE <signal>	The function saved the specified signal to be used in conjunction with the DELTA command
<signal> DELTA min <limit> goto <state>	Transition to specified state if the delta change from the SAVE command is not less than the limit test

Table 3. Fault transitions and redlines.

MS1 > MAH	AS1.3.4 > MAH	AS3 > MAS	AS4 > MAS	PS1 > MAS
MAINSTATUSMON	MAINSTATUSMON	MAINSTATUSMON	MAINSTATUSMON	MAINSTATUSMON
ALLIEDSTATUSMON	ALLIEDSTATUSMON	ALLIEDSTATUSMON	ALLIEDSTATUSMON	ALLIEDSTATUSMON
EMERSS	EMERSS	EMERSS	EMERSS	EMERSS
PT0203 < 1200	PT0203 < 1200 (not as 1)	PT0203 < 1200	PT0203 < 1200	PT0203 < 1200
		PT0207 < 100	PT0451 > 660	PT0451 > 660
MS1 > MAS	AS1 > MAS	TT-364 delta -40	PT0453 > 600	PT0453 > 600
PT0651 > 30	PT0651 > 30	PT0364 < 50	PT0401 > 870	PT0401 > 870
	PT0203 < 1200	PT0651 < 500		PT0651 > 30
			SS1 > MAS	PT0163 > 700
IN1 > MAH		CO1 > MAS	MAINSTATUSMON	PT0001-8 > 320
MAINSTATUSMON	AS2 > MAS	MAINSTATUSMON	ALLIEDSTATUSMON	PT0364 > 900
ALLIEDSTATUSMON	MAINSTATUSMON	ALLIEDSTATUSMON	EMERSS	
EMERSS	ALLIEDSTATUSMON	EMERSS	PT0104 < 250	MCS > MAH
PT0203 < 1200	EMERSS	PT0203 < 1200	PT0203 < 1200	MAINSTATUSMON
PT0202 < 50	PT0203 < 1200	PT0207 < 100	PT0364 > 430, > 900	ALLIEDSTATUSMON
PT0651 < 500	PT0104 < 250	PT0104 < 250	PT0401 > 870	EMERSS
PT0208 < 500	PT0163 < 50	PT0364 > 300	500 > PT0451 > 660	
	PT0101 > 200	PT0163 > 300	135 > PT0453 > 600	MAS > MAH
PSR > MAS	PT0651 > 30	PT0651 < 500	15,150 > PT00018 > 320	timer > 2 sec
MAINSTATUSMON			420 > PT0163 > 700	
ALLIEDSTATUSMON			TT0455-8 > 220	
EMERSS			100 < PT0207 < 100	
PT0203 < 1200			TT0364 delta -40	
PT0651 > 30			PT0651 < 500	

PT(xxxx) - Pressure Transducer, where xxxx is location identifier
TT(xxxx) - Temperature Transducer, where xxxx is location identifier
EMERSS - Emergency shutdown switch
MAINSTATUSMON - activate ASSC fault monitor
ALLIEDSTATUSMON - activate ASC fault monitor
AS(y) - autosafe state, where y is 1-arm, 2-H_2, 3-LO_2, and 4-H_2O
CO1 - cutoff state
IN1 - initialization state
MAS - master abort sequence state
MAH - master abort hold state
MCS - mission complete standby state
MS1 - master standby state
PSR - prestart reset state
SS1 - start state

Table 4. ASSC transition words.

Transition Word	Description
1–6	Six-character name of item causing transition ‘defaul’(t) - default transition ‘TIMER’ - table timer ‘ALLMON’ - ASC abort ‘MANMON’ - ASSC abort ‘PTxxxx’ - name of pressure transducer identified by xxxx causing transition ‘TTxxxx’ - name of temperature tranducer identified by xxxx causing transition
7	Transition condition code ‘a’ - above ‘b’ - below ‘c’ - change (delta) ‘d’ - default ‘f’ - off ‘h’ - main controller over temp of 120 °C ‘m’ - ASC data mismatch ‘n’ - on ‘o’ - ‘outside ‘p’ - ASSC PMC mismatch ‘r’ - ASC not receiving ASSC status ‘s’ - no data saved for delta function ‘t’ - ASC servoamp temperature over limit (40 °C) ‘w’ - wait (timer expires on a wait); refers to a watch dog monitor fail (if MANMON is set) ‘z’ - ASC data word count error
8	Lower limit checked for transition if applicable, otherwise 0. If the transition is an ‘h’, the word contains the ASSC temperature limit.
9	Upper limit checked for transition if applicable, otherwise 0. If the transition condition is a ‘t’ (ASC servoamp temperature over limit), then this word indicates which servoamp is over limit. ‘1’ GH_2 Servoamp ‘2’ LO_2 Servoamp ‘3’ H_2O Servoamp
10	Current value is the transition item, otherwise 0. If the transition condition is an ‘h’, then this word contains the ASSC temperature.
11–14	State in which the transition occurred. These words contain three characters and one number.

Table 5. Abort flags.

Abort flags - bit	Description (all flags are latched)
0	ASSC abort
1	GH_2 valve mismatch abort
2	ASC abort. If this bit is set without any other ASC related bit, then the ASC has not received data from the ASC for at least two consecutive requests.
3	ASC temperature abort
4	LO_2 valve indication mismatch abort
5	H_2O valve indication mismatch abort
6	LO_2 clutch indication mismatch abort
7	H_2O clutch indication mismatch abort
8	GH_2 clutch indication mismatch abort
9	ASC voltage abort
10	ASC ready bit abort
11	ASSC temperature abort

Table 6. Sensor model of start states.

State	Signal		State	Signal	
SS1:2	PT0651 =	600.0	SS1:18	PT0401 =	700.0
SS1:4	PT0651 =	15.0	SS1:20	PT0451 =	580.0
				PT0453 =	500.0
SS1:5	PT0651 =	600.0	SS1:21	PT0104 =	300.0
SS1:8	PT0208 =	720.0	SS1:23	PT0104 =	15.0
	PT0651 =	600.0			
	PT0207 =	200.0			
SS1:9	PT0651 =	600.0	SS1:24	PT0104 =	300.0
SS1:10	PT0207 =	15.0	SS1:28	PT0401 =	777.0
SS1:11	PT0207 =	200.0	SS1:30	PT0651 =	30.0
SS1:13	TT0364 =	70.0	SS1:33	PT0001 =	220.0
				PT0002 =	220.0
				PT0003 =	220.0
				PT0004 =	220.0
				PT0005 =	220.0
				PT0006 =	220.0
				PT0007 =	220.0
				PT0008 =	220.0
				TT0455 =	170.0
				TT0456 =	170.0
				TT0457 =	170.0
				TT0458 =	170.0
SS1:14	PT0207 =	15.0	SS1:35	PT0104 =	300.0
				PT0164 =	500.0
SS1:16	TT0364 =	–200.0	SS1:36	PT0104 =	15.0
	PT0210 =	1313.0		PT0210 =	1313.0

Table 7. Script file.

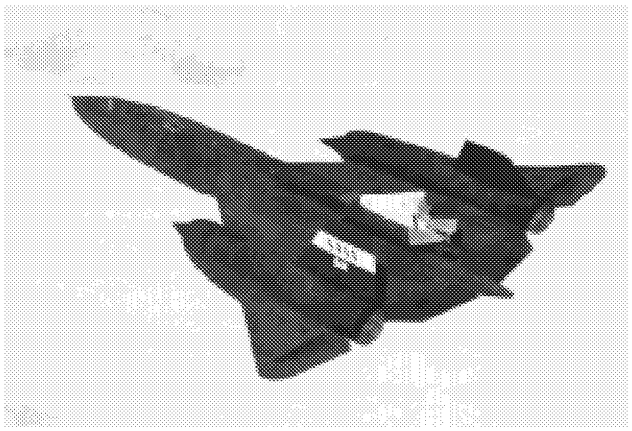
```
# Test redline on PT0001 high (320) in ss1:37
logc
file
do /home/not/acts/ST_HOTFIRE/VnV/Scripts/Start/initsensor
do /home/not/acts/ST_HOTFIRE/VnV/Scripts/Start/prestartinit.scr
# Set test values
mdl
state=SS1:36
PT0001=325
# Select TEA-TEB canister #1
cp
TEATEBSEL=1
# Press the start button
START=1; START=0
# Wait for state MAH:1 (Master Abort Hold)
mdl
imxtst=6000;    - Timeout = 60 seconds
waitst=MAH:1;   - Set wait state to master abort hold
val
lwaitst=1;      - Wait for master abort hold state
# Controller should now be in state MAH:001
prst
# Controller should now be in state MAH:001
# Validation/Verification checks for this test script:
# system goes to MAS in SS1:37?                    _______
# abort is due to redline of PT0001 above 320?     _______
#
# Date:_____  Initials:_____
file
do /home/not/acts/ST_HOTFIRE/VnV/Scripts/Start/msinit
```

Table 8. Log file sample.

Time tag	Status information
61:01:25:634	- NEW STATE - SS1:035
61:01:25:644	- PT0163 - SET TO 500.00
61:01:25:654	- TRANSITION - DEFAUL D 0032 0032 0032 SS1:035
61:01:25:844	- CLOSED SV0101
61:01:25:854	- TRANSITION - TIMER W 0003 0032 0004 SS1:035
61:01:25:854	- NEW STATE - SS1:036
61:01:25:854	- PT0001 - SET TO 325.00
61:01:25:854	- PT0104 - SET TO 15.00
61:01:25:864	- TRANSITION - DEFAULD D 0032 0032 0032 SS1:036
61:01:25:884	- STARTIND - LIGHT OFF
61:01:25:884	- CONABORTIND - LIGHT ON
61:01:25:884	- CLOSED PR0101
61:01:25:884	- CLOSED PV0301
61:01:25:884	- CLOSED CL0101
61:01:25:884	- CLOSED CL0301
61:01:25:884	- OPENED SV0101
61:01:25:884	- OPENED SV0202
61:01:25:884	- CLOSED SV0304
61:01:25:884	- CLOSED SV0401
61:01:25:884	- CLOSED SV0402
61:01:25:894	**- TRANSITION - PT0001 O 0144 2078 2110 SS1:036**
61:01:25:894	- NEW STATE - MAS:001
61:01:25:904	- TRANSITION - DEFAUL D 0032 0032 0032 MAS:001
61:01:25:904	- NEW STATE - MAS:002
61:01:25:914	- TRANSITION - DEFAUL D 0032 0032 0032 MAS:002
61:01:26:114	- OPENED SV0303
61:01:26:134	- TRANSITION - TIMER W 0004 0032 0005 MAS:002
61:01:26:134	- NEW STATE - MAS:003
61:01:26:134	- PT0104 - SET TO 810.00
61:01:26:134	- PT0163 - SET TO 40.00
61:01:26:134	- PT0207 - SET TO 720.00
61:01:26:134	- PT0364 - SET TO 40.00
61:01:26:134	- PT0651 - SET TO 600.00
61:01:26:144	- TRANSITION - DEFAUL D 0032 0032 0032 MAS:003
61:01:26:884	- OPENED SV0205
61:01:26:884	- CLOSED SV0212
61:01:26:894	- TRANSITION - TIMER W 0018 0032 0019 MAS:003
61:01:26:894	- NEW STATE - MAS:004
61:01:26:904	- TRANSITION - DEFAUL D 0032 0032 0032 MAS:004
61:01:28:864	- CLOSED PV0401
61:01:28:864	- CLOSED SV0403
61:01:28:874	- TRANSITION - TIMER W 0054 0032 0055 MAS:004

Table 9. State time comparisons.

FDAS times	SMART times	State
09.19.53.290		SS1:1
09.19.53.300	09:19:53.300	SS1:2
09.19.53.480		SS1:3
09.19.53.490	09:19:53.499	SS1:7
09.19.53.760	09:19:53.765	SS1:8
09.19.53.870	09:19:53.863	SS1:9
09.19.53.880	09:19:54.756	SS1:13
09.19.54.740	09:19:56.008	SS1:14
09.19.55.790		SS1:15
09.19.55.800	09:19:59.871	SS1:16
09.19.59.740	09:19:59.740	SS1:17
09.19.59.750	09:20:00.567	SS1:18
09.20.00.520	09:20:00.533	SS1:19
09.20.02.270	09:20:02.284	SS1:20
09.20.02.550	09:20:02.680	SS1:21
09.20.02.650		SS1:22
09.20.02.660	09:20:02.946	SS1:26
09.20.02.820	09:20:03.472	SS1:27
09.20.03.310	09:20:04.004	SS1:28
09.20.03.800	09:20:04.530	SS1:29
09.20.04.310		SS1:30
09.20.04.320		SS1:31
09.20.04.330	09:20:04.793	SS1:33
09.20.04.630	09:20:04.793	SS1:34
09.20.04.690	09:20:05.027	SS1:35
09.20.04.910	09:20:05.357	SS1:36
09.20.05.240		SS1:37
09.20.05.250	09:20:05.800	SS1:38



EC97-44295-108

Figure 1. The LASRE aircraft.

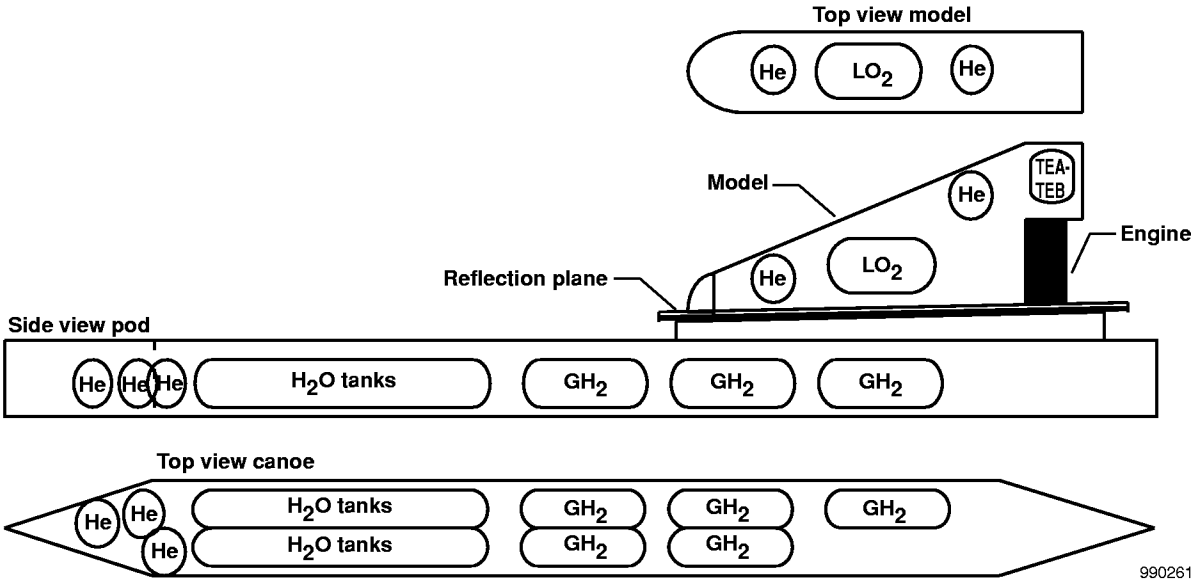


Figure 2. The LASRE feed tank configuration.

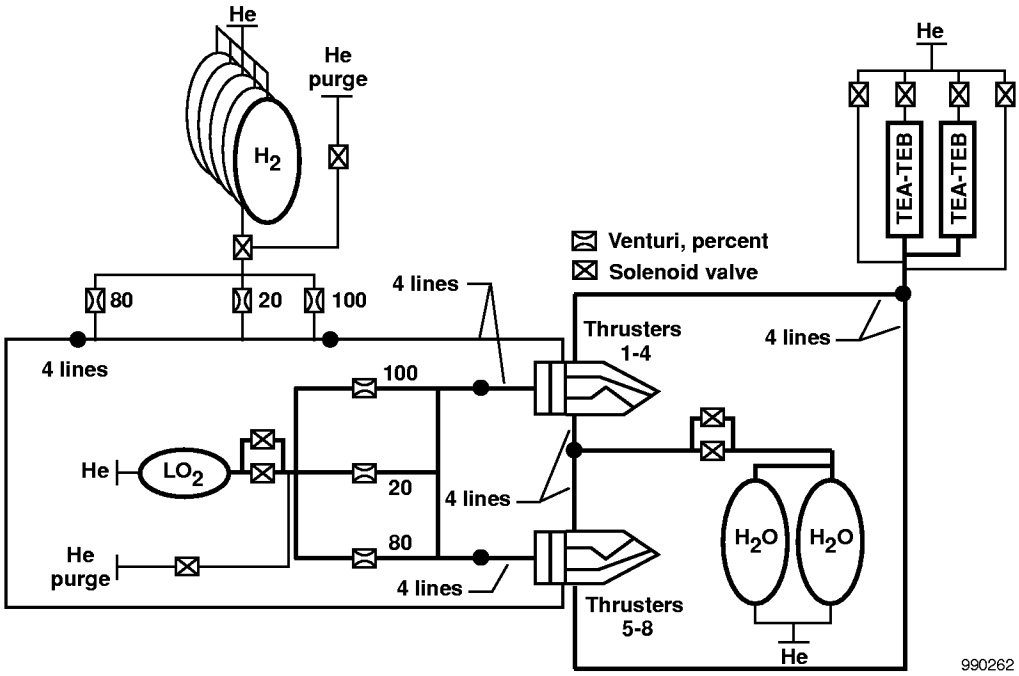


Figure 3. LASRE supply systems valve arrangement.

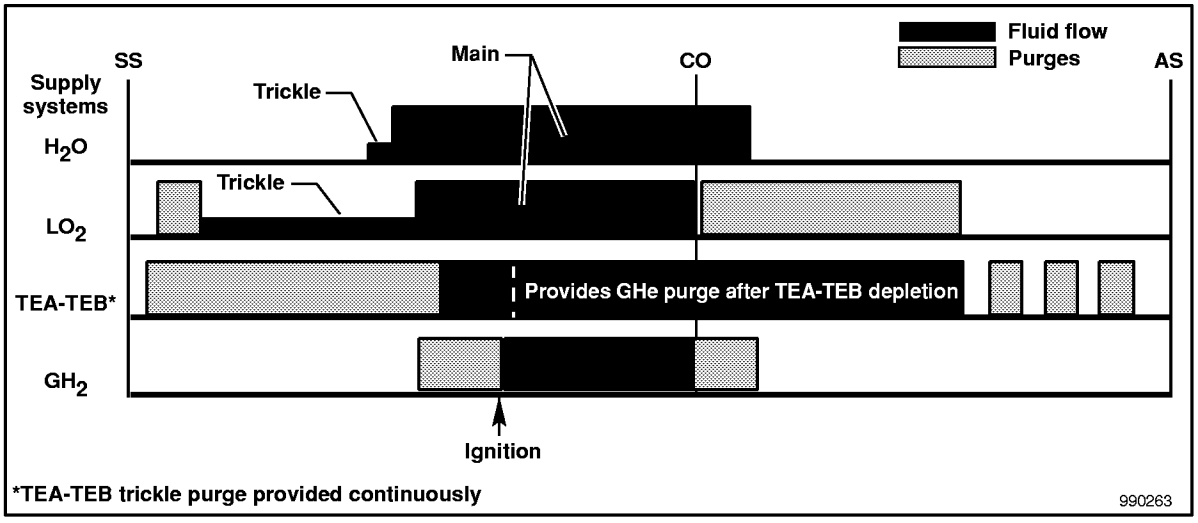


Figure 4. LASRE engine firing sequence.

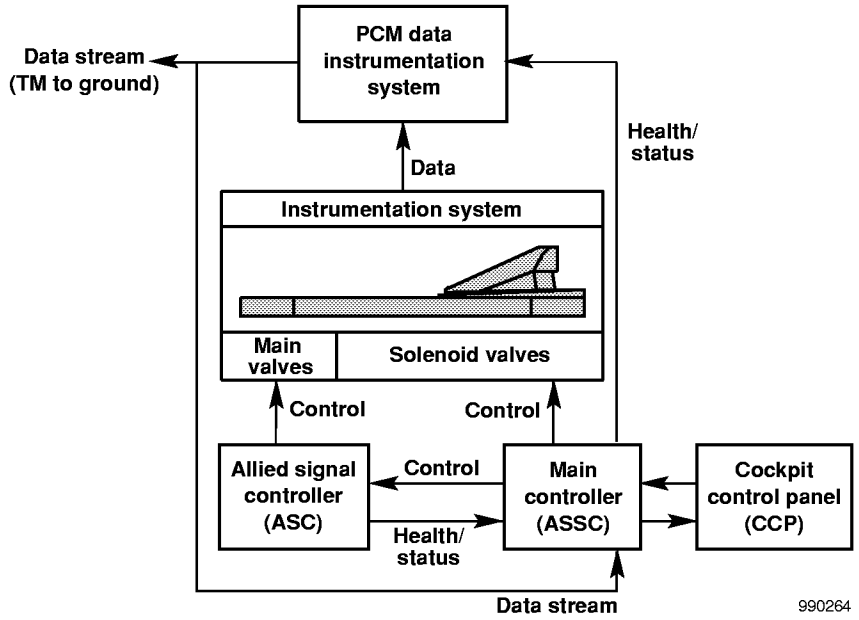


Figure 5. LASRE controller architecture.

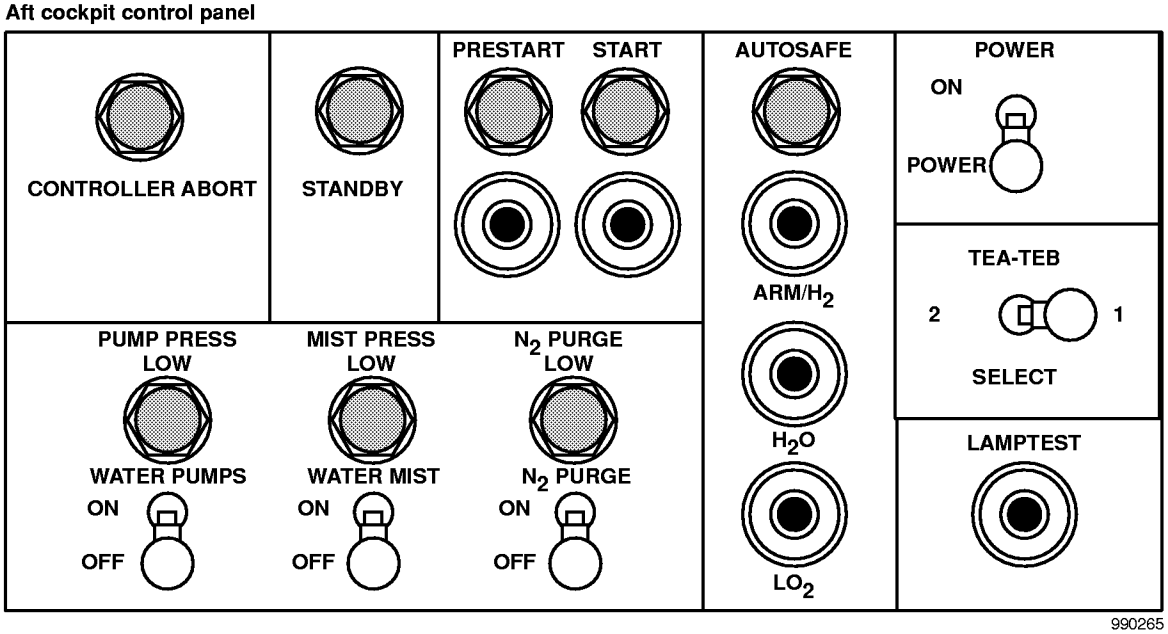


Figure 6. SR-71 Aft cockpit control panel.

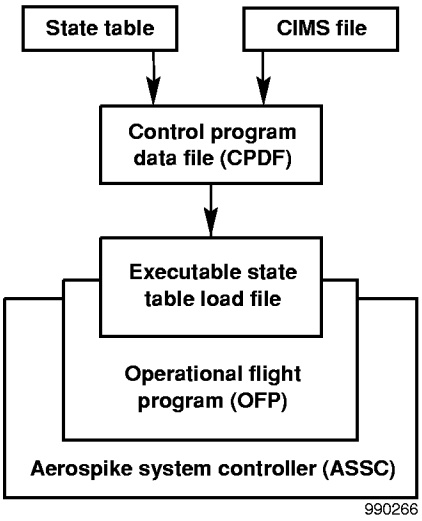


Figure 7. ASSC software modules.

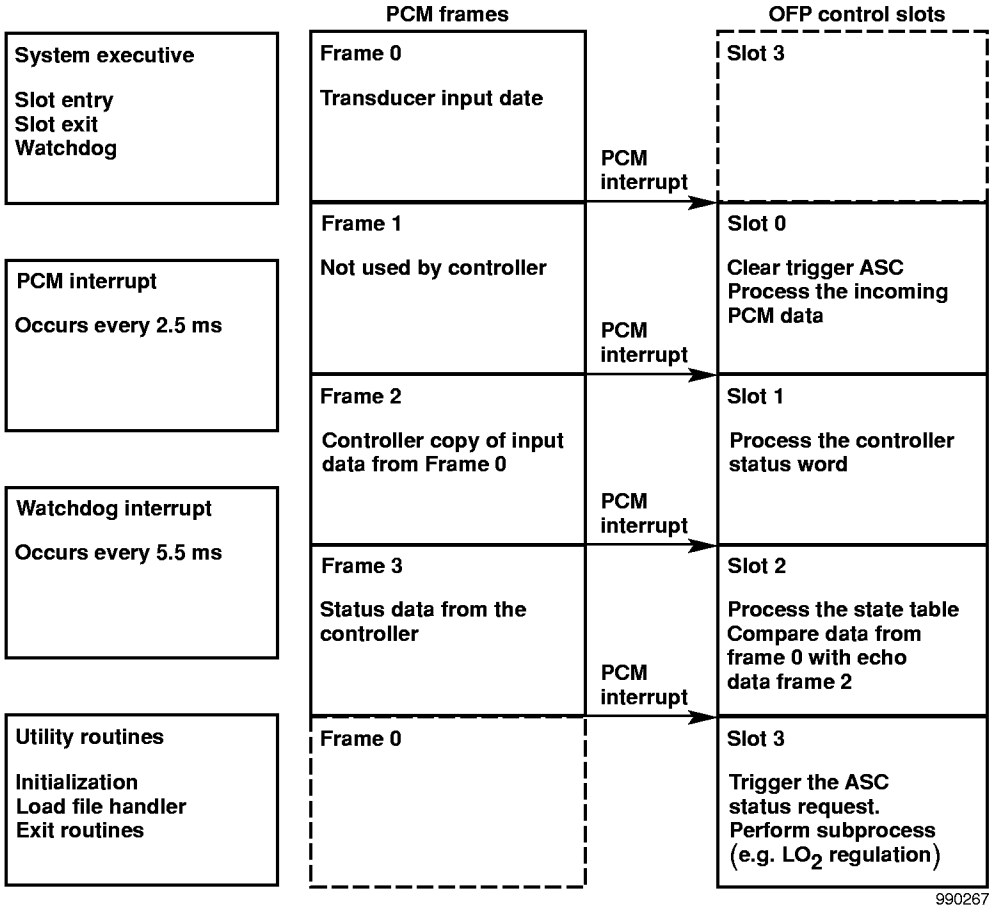


Figure 8. OFP frame interrupts.

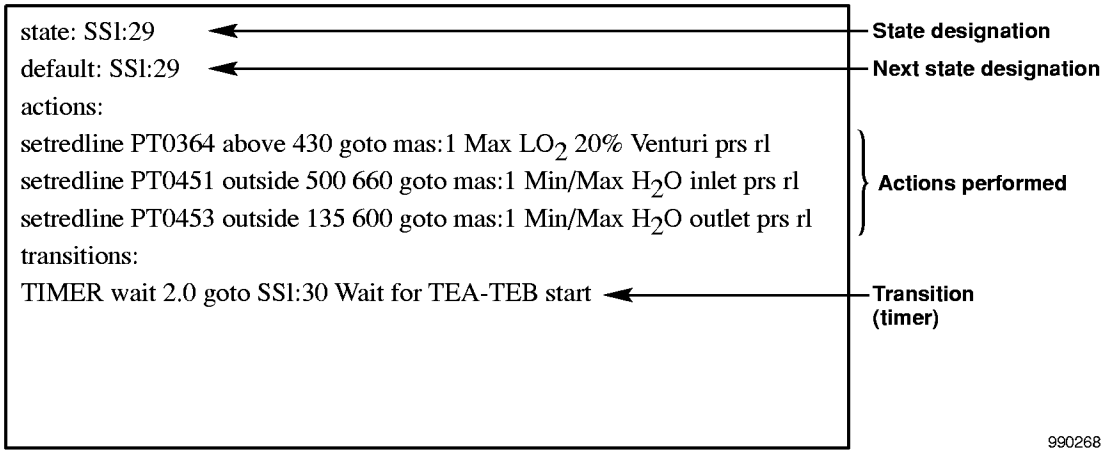


Figure 9. State table example.

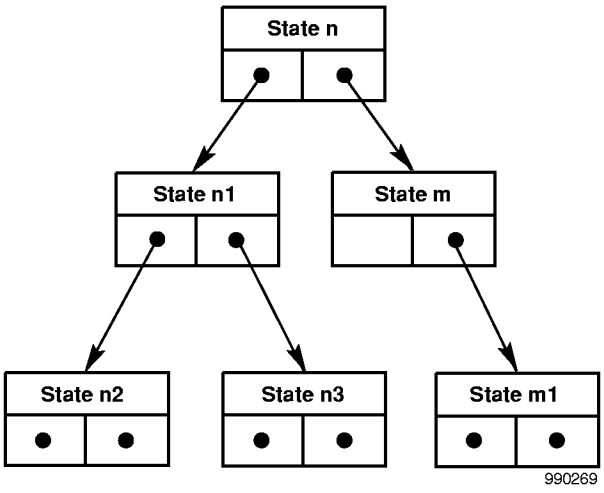


Figure 10. Linked list data structure.

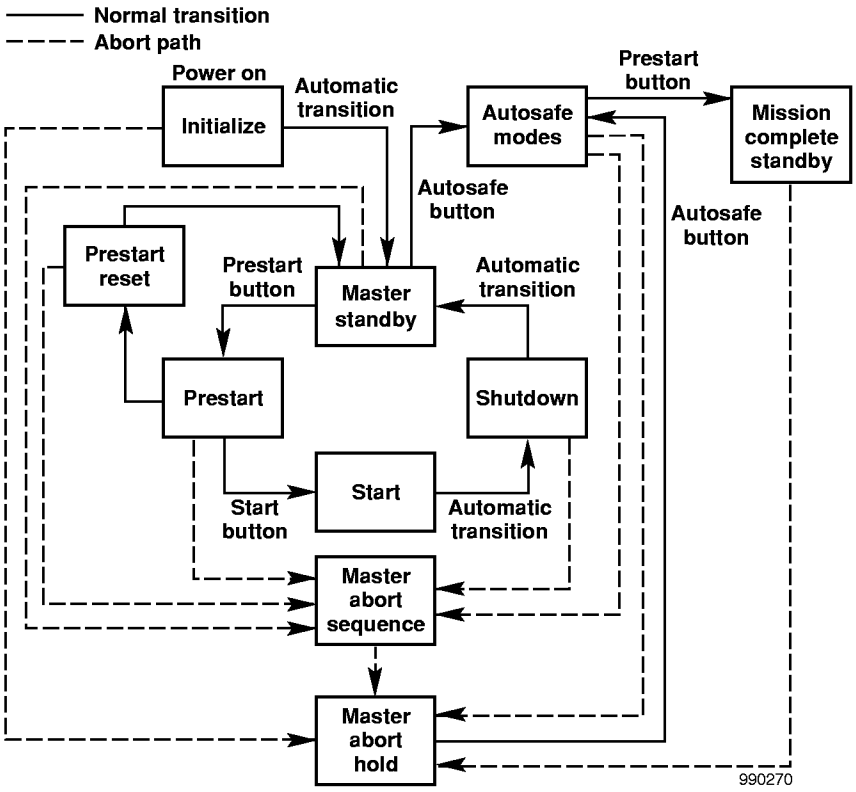


Figure 11. ASSC state transition overview.

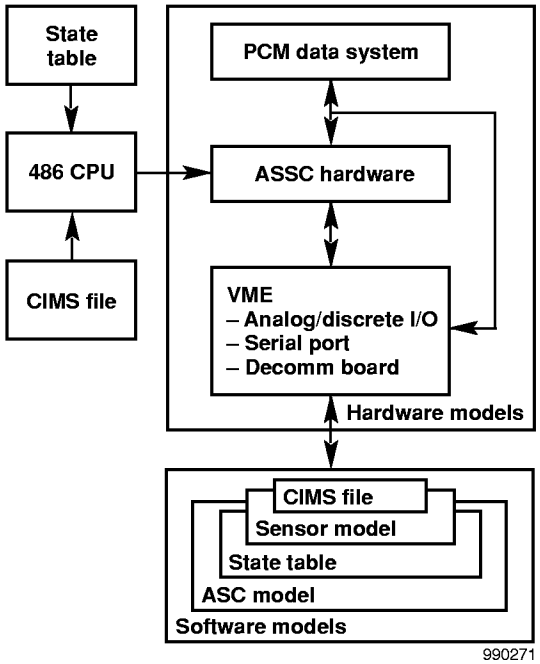


Figure 12. Aerospike controller test system.

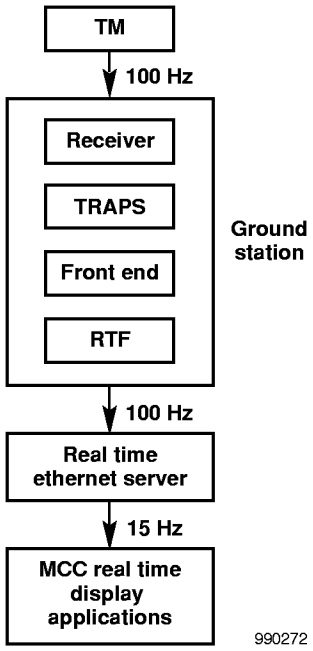


Figure 13. Ground station PCM data flow.

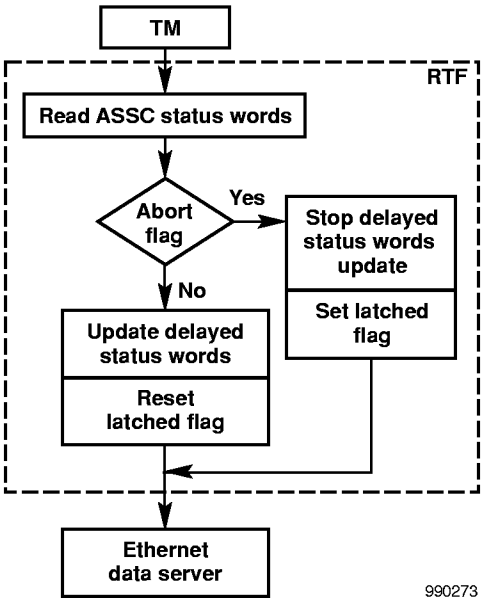


Figure 14. RTF abort latch logic.

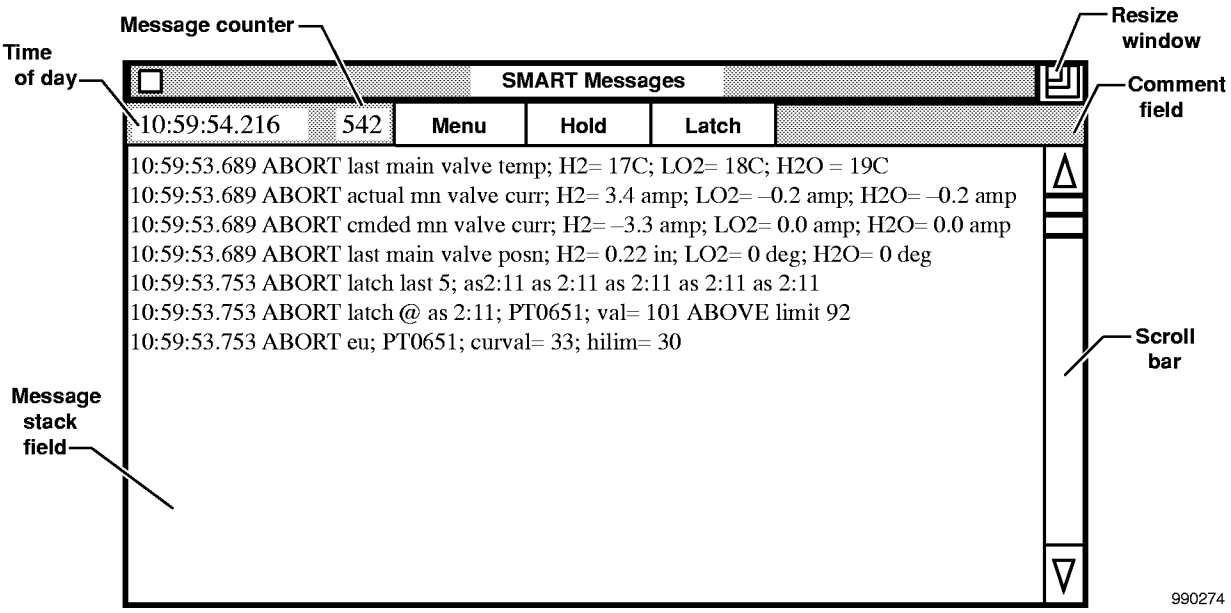


Figure 15. SMART message stack.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1999	Technical Memorandum

4. TITLE AND SUBTITLE

Automated Testing Experience of the Linear Aerospike SR-71 Experiment (LASRE) Controller

5. FUNDING NUMBERS

WU 242-33-02-00-23-00-T15

6. AUTHOR(S)

Richard R. Larson

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

NASA Dryden Flight Research Center
P.O. Box 273
Edwards, California 93523-0273

8. PERFORMING ORGANIZATION REPORT NUMBER

H-2380

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA/TM-1999-206588

11. SUPPLEMENTARY NOTES

Presented at the ITEA Test and Evaluation in the Information Age Symposium, September 21–24, 1999, Atlanta, Georgia, at a poster session.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified—Unlimited
Subject Categories 05, 61, 62

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

System controllers must be fail-safe, low cost, flexible to software changes, able to output health and status words, and permit rapid retest qualification. The system controller designed and tested for the aerospike engine program was an attempt to meet these requirements. This paper describes (1) the aerospike controller design, (2) the automated simulation testing techniques, and (3) the real time monitoring data visualization structure. Controller cost was minimized by design of a single-string system that used an off-the-shelf 486 central processing unit (CPU). A linked-list architecture, with states (nodes) defined in a user-friendly state table, accomplished software changes to the controller. Proven to be fail-safe, this system reported the abort cause and automatically reverted to a safe condition for any first failure. A real time simulation and test system automated the software checkout and retest requirements. A program requirement to decode all abort causes in real time during all ground and flight tests assured the safety of flight decisions and the proper execution of mission rules. The design also included health and status words, and provided a real time analysis interpretation for all health and status data.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Aerospike engine, Automated software testing, Controller architecture, LASRE, Real time analysis, SR-71 experiment	34
	16. PRICE CODE: A03

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited